Part II

Applications in Image Processing & Computer Vision
Chapter 6
Image Representation, Segmentation & Classification

In this chapter, we demonstrate why subspace arrangements can be a very useful class of models for image processing and how the subspace-segmentation techniques may facilitate many important image processing tasks, such as image representation (compression), segmentation, and classification.

6.1 Lossy Image Representation

Researchers in image processing and computer vision have long sought for efficient and sparse representations of images. Except for a few image representations such as fractal-based approaches [?], most existing sparse image representations use an effective linear transformation so that the energy of the (transformed) image will be concentrated in the coefficients of a small set of bases of the transformation. Computing such a representation is typically the first step of subsequent (lossy) compression of the image.\(^1\) The result can also be used for other purposes such as image segmentation,\(^2\) classification, and object recognition.

Most of the popular methods for obtaining a sparse representation of images can be roughly classified into two categories.

---

\(^1\)Which involves further quantization and entropy-coding of the so-obtained representation.

\(^2\)As we will study in the next section.
1. Fixed-Basis Linear Transformations.

Methods of the first category seek to transform all images using a \textit{pre-fixed} linear transformation. Each image is then represented as a superposition of a set of basis functions (specified by the transformation). These methods essentially all evolved from the classical Fourier Transform. One variation of the (discrete) Fourier Transform, the Discrete Cosine Transform (DCT), serves as the core of the JPEG standard \cite{Tang:2014}. Due to the Gibbs' phenomenon, DCT is poor at approximating discontinuities in the imagery signal. Wavelets \cite{Liu:2014, Liu:2015, Liu:2016, Liu:2017} have been developed to remedy this problem and have been shown to be optimal for representing 1-D signals with discontinuities. JPEG-2000 adopted wavelets as its standard. However, because wavelet transforms only deal with 1-D discontinuities, they are not well-suited to represent 2-D singularities along edges or contours. Anisotropic bases such as wedgelets \cite{Liu:2014}, curvelets \cite{Liu:2015}, countourlets \cite{Liu:2016} and bandlets \cite{Liu:2017} have been proposed explicitly to capture different 2-D discontinuities. These x-lets have been shown to be (approximately) optimal for representing objects with singularities along $C^2$-smooth edges.\footnote{Here, “optimality” means that the transformation achieves the optimal asymptotic for approximating the class of functions considered \cite{Tang:2014}.}

However, natural images, especially images that have complex textures and patterns, do not consist solely of discontinuities along $C^2$-smooth edges. This is probably the reason why these edge-based methods do not seem to outperform (separable) wavelets on complex images. More generally, one should not expect that a (fixed) “gold-standard” transformation would work optimally for all images (and signals) in the world. Furthermore, conventional image (or signal) processing methods are developed primarily for gray-scale images. For color images or other multiple-valued images, one has to apply them to each value separately (e.g., one color channel at a time). The strong correlation that is normally present among the multiple values or colors is unfortunately ignored.

2. Adaptive Transformations & Hybrid Models

Methods of the second category aim to identify the optimal (or approximately optimal) representation that is \textit{adaptive} to specific statistics or structures of each image.\footnote{Here, unlike in the case of prefixed transformations, “optimality” means the representation obtained is the optimal one within the class of models considered, in the sense that it minimizes certain discrepancy between the model and the data.} The Karhunen-Loève transform (KLT) or principal component analysis (PCA) \cite{Liu:2016} identifies the optimal principal subspace from the statistical correlation of the imagery data and represents the image as a superposition of the basis of the subspace. In theory, PCA provides the optimal linear sparse representation assuming that the imagery data satisfy a uni-modal distribution. However in reality, this assumption is rarely true. Natural images typically exhibit multi-modal statistics as they usually contain many heterogeneous regions with significantly different geometric structures or statistical characteristics (e.g. Figure 6.2). Heterogeneous
data can be better-represented using a mixture of parametric models, one for each homogeneous subset. Such a mixture of models is often referred to as a hybrid model. Vector quantization (VQ) \cite{1} is a special hybrid model that assumes the imagery data are clustered around many different centers. From the dimension reduction point of view, VQ represents the imagery data with many 0-dimensional (affine) subspaces. This model typically leads to an excessive number of clusters or subspaces.\footnote{Be aware that compared to methods in the first category, representations in the second category typically need additional memory to store the information about the resulting model itself, e.g., the basis of the subspace in PCA, the cluster means in VQ.} The primal sketch model \cite{2} is another hybrid model which represents the high entropy parts of images with Markov random fields \cite{3, 4} and the low entropy parts with sketches. The result is also some kind of a “sparse” representation of the image as superposition of the random fields and sketches. However, the primary goal of primal sketch is not to authentically represent and approximate the original image. It is meant to capture the (stochastic) generative model that produces the image (as random samples). Therefore, this type of models are more suited for image parsing, recognition, and synthesis than approximation and compression. In addition, finding the sketches and estimating the parameters of the random fields are computationally expensive and therefore less appealing for developing efficient image representation and compression schemes.

In this chapter, we would like to show how to combine the benefits of PCA and VQ by representing an image with multiple (affine) subspaces – one subspace for one image segment. The dimension and basis of each subspace are pertinent to the characteristics of the image segment it represents. We call this a hybrid linear model and will show that it strikes a good balance between simplicity and expressiveness for representing natural images.

A Multi-Scale Hybrid Linear Model for Lossy Image Representation.

One other important characteristic of natural images is that they are comprised of structures at many different (spatial) scales. Many existing frequency-domain techniques harness this characteristic \cite{5}. For instance, wavelets, curvelets, and fractals have all demonstrated effectiveness in decomposing the original imagery signal into multiple scales (or subbands). As the result of such a multi-scale decomposition, the structures of the image at different scales (e.g., low v.s. high frequency/entropy) become better exposed and hence can be more compactly represented. The availability of multi-scale structures also significantly reduces the size and dimension of the problem and hence reduces the overall computational complexity.

Therefore, in this chapter we introduce a new approach to image representation by combining the hybrid paradigm and the multi-scale paradigm. The result is a multi-scale hybrid linear model which is based on an extremely simple concept: Given an image, at each scale level of its down-sample pyramid, fit the (residual)
image by a (multiple-subspace) hybrid linear model. Compared to the single-scale hybrid linear model, the multi-scale scheme can reduce not only the size of the resulting representation but also the overall computational cost. Surprisingly, as we will demonstrate, such a simple scheme is able to generate representations for natural images that are more compact, even with the overhead needed to store the model, than most state-of-the-art representations, including DCT, PCA, and wavelets.

### 6.1.1 A Hybrid Linear Model

In this section we introduce and examine the hybrid linear model for image representation. The relationship between hybrid linear models across different spatial scales will be discussed in Section 6.1.2.

An image \( I \) with width \( W \), height \( H \), and \( c \) color channels resides in a very high-dimensional space \( \mathbb{R}^{WH \times c} \). We may first reduce the dimension by dividing the image into a set of non-overlapping blocks of size \( b \times b \). Each block of size \( b \times b \) is then stacked into a vector \( x_i \in \mathbb{R}^D \), where \( D = b^2 c \) is the dimension of the ambient space. For example, if \( c = 3 \) and \( b = 2 \), then \( D = 12 \). In this way, the image \( I \) is converted to a set of vectors \( \{x_i \in \mathbb{R}^D\}_{i=1}^N \), where \( N = WH/b^2 \) is the total number of vectors.

Borrowing ideas from existing unsupervised learning paradigms, it is tempting to assume the imagery data \( \{x_i\} \) are random samples from a (non-singular) probability distribution or noisy samples from a smooth manifold. As the distribution or manifold can be very complicated, a common approach is to infer a best approximation within a simpler class of models for the distributions or manifolds. The “optimal” model is then the one that minimizes certain distance to the true model. Different choices of model classes and distance measures have led to many different learning algorithms developed in machine learning, pattern recognition, computer vision, and image processing. The most commonly adopted distance measure, for image compression, is the Mean Square Error (MSE) between the original image \( I \) and approximated image \( \hat{I} \),

\[
\epsilon_I^2 = \frac{1}{WHc} \|\hat{I} - I\|^2. \tag{6.1}
\]

Since we will be approximating the (block) vectors \( \{x_i\} \) rather than the image pixels, in the following derivation, it is more convenient for us to define the Mean Square Error (MSE) per vector which is different from \( \epsilon_I^2 \) by a scale,

\[
\epsilon^2 = \frac{1}{N} \sum_{i=1}^N \|\hat{x}_i - x_i\|^2 = \frac{b^2}{WH} \sum_{i=1}^N \|\hat{x}_i - x_i\|^2 = \frac{b^2}{WH} \|\hat{I} - I\|^2 = (b^2c)\epsilon_I^2. \tag{6.2}
\]

---

\(^6\)Therefore, \( b \) needs to be a common divisor of \( W \) and \( H \).
The Peak Signal to Noise Ratio (PSNR) of the approximated image is defined as,

\[ \text{PSNR} = -10 \log \sigma_{\hat{x}}^2 = -10 \log \frac{\sigma^2}{2^{24}}. \] (6.3)

**Linear Models.**

If we assume that the vectors \( \mathbf{x} \) are drawn from an anisotropic Gaussian distribution or a linear subspace, the optimal model subject to a given PSNR can be inferred by Principal Component Analysis (PCA) [Pearson, 1901, Hotelling, 1933, Jollife, 2002] or equivalently the Karhunen-Loève Transform (KLT) [?]. The effectiveness of such a linear model relies on the assumption that, although \( D \) can be large, all the vectors \( \mathbf{x} \) may lie in a subspace of a much lower dimension in the ambient space \( \mathbb{R}^D \). Figure 6.1 illustrates this assumption.

![Figure 6.1](image)

Figure 6.1. In a linear model, the imagery data vectors \( \{\mathbf{x}_i \in \mathbb{R}^D\} \) reside in an (affine) subspace \( S \) of dimension \( d \ll D \).

Let \( \bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i \) be the mean of the imagery data vectors, and \( \mathbf{X} = [\mathbf{x}_1 - \bar{\mathbf{x}}, \mathbf{x}_2 - \bar{\mathbf{x}}, \ldots, \mathbf{x}_N - \bar{\mathbf{x}}] = U \Sigma V^T \) be the SVD of the mean-subtracted data matrix \( \mathbf{X} \). Then all the vectors \( \mathbf{x}_i \) can be represented as a linear superposition:

\[ \mathbf{x}_i = \bar{\mathbf{x}} + \sum_{j=1}^{D} \alpha_j \mathbf{v}_j, \quad i = 1, \ldots, N, \]

where \( \{\mathbf{v}_j\}_{j=1}^{D} \) are just the columns of the matrix \( U \).

The matrix \( \Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_D) \) contains the ordered singular values \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_D \). It is well known that the optimal linear representation of \( \mathbf{x}_i \) subject to the MSE \( \epsilon^2 \) is obtained by keeping the first \( d \) (principal) components

\[ \hat{\mathbf{x}}_i = \bar{\mathbf{x}} + \sum_{k=1}^{d} \alpha_k \mathbf{v}_k, \quad i = 1, \ldots, N, \] (6.4)

where \( d \) is chosen to be

\[ d = \min(k), \quad \text{s.t.} \quad \frac{1}{N} \sum_{i=k+1}^{D} \sigma_i^2 \leq \epsilon^2. \] (6.5)

\(^7\)The peak value of the imagery data is normalized into 1.
The model complexity of the linear model, denoted as $\Omega$, is the total number of coefficients needed for representing the model $\{\alpha_i, \phi_k, \bar{x}\}$ and subsequently a lossy approximation $\hat{I}$ of the image $I$. It is given by

$$\Omega(N, d) = Nd + \varnothing(D - d + 1),$$

(6.6)

where the first term is the number of coefficients $\{\alpha_i\}$ to represent $\{\bar{x}_i - \bar{x}\}_i=1^N$ with respect to the basis $\Phi = \{\phi_k\}_{k=1}^D$ and the second term is the number of Grassmannian coordinates$^8$ needed for representing the basis $\Phi$ and the mean vector $\bar{x}$. The second term is often called overhead.$^9$ Notice that the original set of vectors $\{x_i\}$ contain $ND$ coordinate entries. If $\Omega \ll ND$, the new representation, although lossy, is more compact. The search for such a compact representation is at the heart of any (lossy) image compression method. When the image $I$ is large and the block size $b$ is small, $N$ will be much larger than $D$ so that the overhead will be much smaller than the first term. However, in order to compare fairly with other methods, in the subsequent discussions and experiments, we always count the total number of coefficients needed for the representation, including the overhead.

**Hybrid Linear Models.**

The linear model is very efficient when the target manifold or distribution function is indeed unimodal. However, if the image $I$ contains several heterogeneous regions $\{I_j\}_{j=1}^n$, the data vectors $x_i$ can be samples from a collection of subspaces of possibly different dimensions or from a mixture of multiple (Gaussian) distributions. Figure 6.2 shows the first three principal components of the data.

![Figure 6.2](image.png)

Figure 6.2. Left: The baboon image. Right: The coordinates of each dot are the first three principal components of the vectors $x_i$. There is a clear multi-modal structure in the data.

$^8$Notice that to represent a $d$-dimensional subspace in a $D$-dimensional space, we only need to specify a basis of $d$ linearly independent vectors for the subspace. We may stack these vectors as rows of a $d \times D$ matrix. Any nonsingular linear transformation of these vectors span the same subspace. Thus, without loss of generality, we may assume that the matrix is of the normal form $[I_{d \times d}; G]$ where $G$ is a $d \times (D - d)$ matrix consisting of the so-called Grassmannian coordinates.

$^9$Notice that if one uses a pre-chosen basis such as discrete Fourier transform, discrete cosine transform (JPEG), and wavelets (JPEG-2000), there is no such overhead.
vector \( \mathbf{x}_i \) (as dots in \( \mathbb{R}^3 \)) of an image. Note the clear multi-modal characteristic in the data.

Suppose that a natural image \( I \) can be segmented into \( n \) disjoint regions \( I = \bigcup_{j=1}^n I_j \) with \( I_j \cap I_{j'} = \emptyset \) for \( j \neq j' \). In each region \( I_j \), we may assume the linear model (6.4) is valid for the subset of vectors \( \{ \mathbf{x}_{j,i} \}_{i=1}^{N_j} \) in \( I_j \):

\[
\hat{\mathbf{x}}_{j,i} = \overline{\mathbf{x}}_j + \sum_{k=1}^{d_j} \alpha^k_{i,j,k}, \quad i = 1, \ldots, N_j.
\]  

(6.7)

Intuitively, the hybrid linear model can be illustrated by Figure 6.3.

![Figure 6.3](image.png)

Figure 6.3. In hybrid linear models, the imagery data vectors \( \{ \mathbf{x}_i \} \) reside in multiple (affine) subspaces which may have different dimensions.

As in the linear model, the dimension \( d_j \) of each subspace is determined by a common desired MSE \( \epsilon^2 \) using equation (6.5). The model complexity, i.e., the total number of coefficients needed for representing the hybrid linear model \( \{ \phi_{j,k}, \hat{\mathbf{x}}_{j,i} \} \) is\(^{10}\)

\[
\Omega = \Omega(N_1, d_1) + \cdots + \Omega(N_n, d_n) = \sum_{j=1}^n \left( N_j d_j + d_j(D - d_j + 1) \right).
\]

(6.8)

Notice that \( \Omega \) is similar to the effective dimension (ED) of the hybrid linear representation defined in [9]. Thus, finding a representation that minimizes \( \Omega \) is the same as minimizing the effective dimension of the imagery data set.\(^{11}\)

Instead, if we model the union of all the vectors \( \bigcup_{j=1}^n \{ \mathbf{x}_{j,i} \}_{i=1}^{N_j} \) with a single subspace (subject to the same MSE), the dimension of the subspace in general needs to be \( d = \min \{ d_1 + \cdots + d_n, D \} \). It is easy to verify from the definition (6.6) that under reasonable conditions (e.g., \( n \) is bounded from being too large),

\[^{10}\text{We also needs a very small number of binary bits to store the membership of the vectors. But those extra bits are insignificant comparing to } \Omega \text{ and often can be ignored.}\]

\[^{11}\text{In fact, the minimal } \Omega \text{ can also be associated to the Kolmogorov entropy or to the minimum description length (MDL) of the imagery data.}\]
we have

\[ \Omega(N, d) > \Omega(N_1, d_1) + \cdots + \Omega(N_n, d_n). \] (6.9)

Thus, if a hybrid linear model can be identified for an image, the resulting representation will in general be much more compressed than that with a single linear or affine subspace. This will also be verified by experiments on real images in Section 6.1.3.

However, such a hybrid linear model alone is not able to generate a representation that is as compact as that by other competitive methods such as wavelets. There are at least two aspects in which the above model can be further improved. Firstly, we need to further reduce the negative effect of overhead by incorporating a pre-projection of the data onto a lower dimensional space. Secondly, we need to implement the hybrid linear model in a multi-scale fashion. We will discuss the former aspect in the remainder of this section and leave the issues with multi-scale implementation to the next section.

**Dimension Reduction via Projection.**

In the complexity of the hybrid linear model (6.8), the first term is always smaller than that of the linear model (6.6) because \( d_j \leq d \) for all \( j \) and \( \sum_{j=1}^{n} N_j = N \). The second overhead term however can be larger than in that of the linear model (6.6) because the bases of multiple subspaces now must be stored. We here propose a method to further reduce the overhead by separating the estimation of the hybrid model into two steps.

In the first step, we may project the data vectors \( \{x_i\} \) onto a lower-dimensional subspace (e.g., via PCA) so as to reduce the dimension of the ambient space from \( D \) to \( D' \). The justification for such a subspace projection has been discussed earlier in Section 3.2.2. Here, the dimension \( D' \) is chosen to achieve an MSE \( \frac{1}{2} \epsilon^2 \). The data vectors in the lower ambient space \( \mathbb{R}^{D'} \) are denoted as \( \{x'_i\} \). In the second step, we identify a hybrid linear model for \( \{x'_i\} \) within the lower-dimension ambient space \( \mathbb{R}^{D'} \). In each subspace, we determine the dimension \( d_j \) subject to the MSE \( \frac{1}{2} \epsilon^2 \). The two steps combined achieve an overall MSE \( \epsilon^2 \), but they can actually reduce the total model complexity to

\[ \Omega = \sum_{j=1}^{n} (N_j d_j + d_j(D' - d_j + 1)) + D(D' + 1). \] (6.10)

This \( \Omega \) will be smaller than the \( \Omega \) in equation (6.8) because \( D' \) is smaller than \( D \). The reduction of the ambient space will also make the identification of the hybrid linear model (say by GPCA) much faster.

If the number of subspaces, \( n \), is given, algorithms like GPCA or EM can always find a segmentation. The basis \( \{\phi_{j,k}\} \) and dimension \( d_j \) of each subspace are determined by the desired MSE \( \epsilon^2 \). As \( n \) increases, the dimension of the subspaces may decrease, but the overhead required to store the bases may increase. The optimal \( n^* \) therefore can be found recursively by minimizing \( \Omega \) for different \( n \)’s, as shown in Figure 6.4. From our experience, we found that \( n \) is typically in the
range from 2 to 6 for natural images, especially in a multi-scale implementation that we will introduce next.

Algorithm 6.1 describes the pseudocode for estimating the hybrid linear model of an image \( I \), in which the \( \text{SubspaceSegmentation}(\cdot) \) function is implemented (for the experiments in this chapter) using the GPCA algorithm given in earlier chapters. But it can also be implemented using EM or other subspace segmentation methods.

**Algorithm 6.1 (Hybrid Linear Model Estimation).**

1: \textbf{function} \( \hat{l} = \text{HybridLinearModel}(I, \epsilon^2) \)
2: \{\( x_i \)\} = \text{StackImageIntoVectors}(I);
3: \{\( x_i' \), \( \phi_k \), \( \alpha_k' \)\} = \text{PCA}(\{\( x_i - \bar{x} \)\}, \frac{\epsilon^2}{2});
4: \textbf{for each} possible \( n \) \textbf{do}
5: \{\( x_{j,i}' \)\} = \text{SubspaceSegmentation}(\{\( x_i' \), \( n \)\});
6: \{\( x_{j,i}' \), \( \phi_{j,k} \), \( \alpha_{j,k}' \)\} = \text{PCA}(\{\( x_{j,i}' - \bar{x}' \)\}, \frac{\epsilon^2}{2});
7: compute \( \Omega_n \);
8: \textbf{end for}
9: \( \Omega_{opt} = \min(\Omega_n) \);
10: \( \hat{l} = \text{UnstackVectorsIntoImage}(\{\( \hat{x}_{j,i}' \)\} \text{ with } \Omega_{opt}) \);
11: \textbf{output} \{\( \alpha_k' \), \( \phi_k \), \( \bar{x}' \), \( \alpha_{j,k}' \), \( \phi_{j,k} \), \( \bar{x}_{j,i}' \)\} \text{ with } \Omega_{opt};
12: \textbf{return} \( \hat{l} \).

**Example 6.1 (A Hybrid Linear Model for the Gray-Scale Barbara Image).** Figure 6.5 and Figure 6.6 show intuitively a hybrid linear model identified for the \( 8 \times 8 \) blocks of the standard \( 512 \times 512 \) gray-scale Barbara image. The total number of blocks is \( N = 4,096 \). The GPCA algorithm identifies three subspaces for these blocks (for a given error tolerance), as shown in Figure 6.5. Figure 6.6 displays the three sets of bases for the three
Figure 6.5. The segmentation of the 4,096 image blocks from the Barbara image. The image (left) is segmented into three groups (right three). Roughly speaking, the first subspace contains mostly image blocks with homogeneous textures; the second and third subspaces contain blocks with textures of different spatial orientations and frequencies.

subspaces identified, respectively. It is worth noting that these bases are very consistent with the textures of the image blocks in the respective groups.

Figure 6.6. The three sets of bases for the three subspaces (of blocks) shown in Figure 6.5, respectively. One row for one subspace and the number of base vectors (blocks) is the dimension of the subspace.

6.1.2 Multi-Scale Hybrid Linear Models

There are at least several reasons why the above hybrid linear model needs further improvement. Firstly, the hybrid linear model treats low frequency/entropy regions of the image in the same way as the high frequency/entropy regions, which is inefficient. Secondly, by treating all blocks the same, the hybrid linear model fails to exploit stronger correlations that typically exist among adjacent image blocks.\footnote{For instance, if we take all the $b \times b$ blocks and scramble them arbitrarily, the scrambled image would be fit equally well by the same hybrid linear model for the original image.} Finally, estimating the hybrid linear model is computationally expensive when the image is large. For example, we use 2 by 2 blocks, a 512 by 512 color image will have $M = 65,536$ data vectors in $\mathbb{R}^{12}$. Estimating a hybrid linear model for such a huge number of vectors is difficult (if not impossible) on a regular PC. In this section, we introduce a multi-scale hybrid linear representation which is able to resolve the above issues.
The basic ideas of multi-scale representations such as the Laplacian pyramid \cite{cohen1989six} have been exploited for image compression for decades (e.g., wavelets, sub-band coding). A multi-scale method will give a more compact representation because it encodes low frequency/entropy parts and high frequency/entropy parts separately. The low frequency/entropy parts are invariant after low-pass filtering and down-sampling, and can therefore be extracted from the much smaller down-sampled image. Only the high frequency/entropy parts need to be represented at a level of higher resolution. Furthermore, the stronger correlations among adjacent image blocks will be captured in the down-sampled images because every four images blocks are merged into one block in the down-sampled image. At each level, the number of imagery data vectors is one fourth of that at one level above. Thus, the computational cost can also be reduced.

We now introduce a multi-scale implementation of the hybrid linear model. We use the subscript \( l \) to indicate the level in the pyramid of down-sampled images.\textsuperscript{13} The finest level (the original image) is indicated by \( l = 0 \). The larger is \( l \), the coarser is the down-sampled image. We denote the highest level to be \( l = L \).

**Pyramid of Down-Sampled Images.**

First, the level-\( l \) image \( I_l \) passes a low-pass filter \( F_1 \) (averaging or Gaussian filter, etc) and is down-sampled by 2 to get a coarser version image \( \hat{I}_{l+1} \):

\[
I_{l+1} \doteq F_1(I_l) \downarrow 2, \quad l = 0, ..., L - 1. \tag{6.11}
\]

The coarsest level-\( L \) image \( I_L \) is approximated by \( \hat{I}_L \) using a hybrid linear model with the MSE \( \epsilon_L^2 \). The number of coefficients needed for the approximation is \( \Omega_L \).

**Pyramid of Residual Images.**

At all other level-\( l, l = 0, ..., L - 1 \), we do not need to approximate the down-sampled image \( I_l \) because it has been roughly approximated by the image at level-\( (l + 1) \) upsampled by 2. We only need to approximate the residual of this level, denoted as \( I'_l \):

\[
I'_l \doteq I_l - F_2(\hat{I}_{l+1}) \uparrow 2, \quad l = 0, ..., L - 1, \tag{6.12}
\]

where the \( F_2 \) is an interpolation filter. Each of these residual images \( I'_l, l = 0, ..., L - 1 \) is approximated by \( \hat{I}'_l \) using a hybrid linear model with the MSE \( \epsilon'_L \). The number of coefficients needed for the approximation is \( \Omega'_l \), for each \( l = 0, ..., L - 1 \).

**Pyramid of Approximated Images.**

The approximated image at the level-\( l \) is denoted as \( \hat{I}_l \):

\[
\hat{I}_l \doteq \hat{I}'_l + F_2(\hat{I}_{l+1}) \uparrow 2, \quad l = 0, ..., L - 1. \tag{6.13}
\]

The Figure 6.7 shows the structure of a three-level (\( L = 2 \)) approximation of the

\textsuperscript{13}This is not to be confused with the subscript \( j \) used to indicate different segments of an image.
The total number of coefficients needed for the representation will be

$$\Omega = \sum_{l=0}^{L} \Omega_l. \quad (6.14)$$

Figure 6.7. Laplacian pyramid of the multi-scale hybrid linear model.

image $I$. Only the hybrid linear models for $\hat{I}_2$, $\hat{I}_1$, and $\hat{I}_0$, which are approximation for $I_2$, $I_1$, and $I_0$ respectively, are needed for the final representation of the image. Figure 6.8 shows the $I_2$, $I_1$, and $I_0$ for the baboon image.

Figure 6.8. Multi-scale representation of the Baboon image. Left: The coarsest level image $I_2$. Middle: The residual image $I_1$. Right: The residual image $I_0$. The data at each level are modeled as the hybrid linear models. The contrast of the middle and right images has been adjusted so that they are visible.
MSE Threshold at Different Scale Levels.

The MSE thresholds at different levels should be different but related because the up-sampling by 2 will enlarge 1 pixel at level-(l + 1) into 4 pixels at level-l. If the MSE of the level-(l + 1) is $\epsilon_{l+1}^2$, the MSE of the level-l after the up-sampling will become $4\epsilon_{l+1}^2$. So the MSE thresholds of level-(l + 1) and level-l are related as

$$
\epsilon_{l+1}^2 = \frac{1}{4} \epsilon_l^2, \quad l = 0, \ldots, L - 1.
$$

(6.15)

Usually, the user will only give the desired MSE for the approximation of original image which is $\epsilon^2$. So we have

$$
\epsilon_l^2 = \frac{1}{4^l} \epsilon^2, \quad l = 0, \ldots, L.
$$

(6.16)

Vector Energy Constraint at Each Level.

At each level-l, $l = 0, \ldots, L - 1$, not all the vectors of the residual need to be approximated. We only need to approximate the (block) vectors $\{x_i\}$ of the residual image $I'_l$ that satisfy the following constraint:

$$
||x'_i||^2 > \epsilon_l^2.
$$

(6.17)

In practice, the energy of most of the residual vectors is close to zero. Only a small portion of the vectors at each level-l need to be modeled (e.g. Figure 6.9). This property of the multi-scale scheme not only significantly reduces the overall representation complexity $\Omega$ but also reduces the overall computational cost as the number of data vectors processed at each level is much less than those of the original image. In addition, for a single hybrid linear model, when the image size increases, the computational cost will increase in proportion to the square of the image size. In the multi-scale model, if the image size increases, we can...
correspondingly increase the number of levels and the complexity increases only linearly in proportion to the image size.

The overall process of estimating the multi-scale hybrid linear model can be written as the recursive pseudocode in Algorithm 6.2.

Algorithm 6.2 (Multi-Scale Hybrid Linear Model Estimation).

1: function $\hat{I} = \text{MultiscaleModel}(I, level, \varepsilon^2)$
2: if $level < \text{MAXLEVEL}$ then
3: $I_{\text{down}} = \text{Downsample}(F_1(I))$
4: $I_{\text{nextlevel}} = \text{MultiscaleModel}(I_{\text{down}}, level + 1, \frac{1}{4}\varepsilon^2)$
5: end if
6: if $level = \text{MAXLEVEL}$ then
7: $I' = I$
8: else
9: $I_{up} = F_2(\text{Upsample}(I_{\text{nextlevel}}))$
10: $I' = I - I_{up}$
11: end if
12: $\hat{I}' = \text{HybridLinearModel}(I', \varepsilon^2)$;
13: return $I_{up} + I'$.

6.1.3 Experiments and Comparisons

Comparison of Different Lossy Representations.

The first experiment is conducted on two standard images commonly used to compare image compression schemes: the $480 \times 320$ hill image and the $512 \times 512$ baboon image shown in Figure 6.10. We choose these two images because they are representative of two different types of images. The hill image contains large low frequency/entropy regions and the baboon image contains mostly high frequency/entropy regions. The size of the blocks $b$ is chosen to be 2 and the level of the pyramid is 3 – we will test the effect of changing these parameters in subsequent experiments. In Figure 6.11, the results of the multi-scale hybrid linear model are compared with several other commonly used image representations including DCT, PCA/KLT, single-scale hybrid linear model and Level-3 (Daubechies) biorthogonal 4.4 wavelets (adopted by JPEG-2000). The $x$-axis of the figures is the ratio of coefficients (including the overhead) kept for the representation, which is defined as,

$$\eta = \frac{\Omega}{WHc}.$$  \hspace{1cm} (6.18)

The $y$-axis is the PSNR of the approximated image defined in equation (6.3). The
6.1. Lossy Image Representation

Figure 6.10. Testing images: the hill image (480 × 320) and the baboon image (512 × 512).

Figure 6.11. Left: Comparison of several image representations for the hill image. Right: Comparison for the baboon image. The multi-scale hybrid linear model achieves the best PSNR among all the methods for both images. Figure 6.12 shows the two recovered images using the same amount of coefficients for the hybrid linear model and the wavelets. Notice that in the area around the whiskers of the baboon, the hybrid linear model preserves the detail of the textures better than the wavelets. But the multiscale hybrid linear model produces a slight block effect in the smooth regions.

We have tested the algorithms on a wide range of images. We will summarize the observations in Section 6.1.4.

Effect of the Number of Scale Levels.

The second experiment shown in Figure 6.13 compares the multi-scale hybrid linear representation with wavelets for different number of levels. It is conducted on the hill and baboon image with 2 by 2 blocks. The performance increases while the number of levels is increased from 3 to 4. But if we keep increasing the number
of levels to 5, the level-5 curves of both wavelets and our method (which are not shown in the figures) coincide with the level-4 curves. The performance cannot improve any more because the down-sampled images in the fifth level are so small that it is hard to be further compressed. Only when the image is large, can we use more levels of down-sampling to achieve a more compressed representation.

Effect of the Block Size.

The third experiment shown in Figure 6.14 compares the multi-scale hybrid linear models with different block sizes from $2 \times 2$ to $16 \times 16$. The dimension of the ambient space of the data vectors $x$ ranges from 12 to 192 accordingly. The testing image is the baboon image and the number of down-sampling levels is 3. For large blocks, the number of data vectors is small but the dimension of the
subspaces is large. So the overhead would be large and seriously degrade the performance. Also the block effect will be more obvious when the block size is large. This experiment shows that 2 is the optimal block size, which also happens to be compatible with the simplest down-sampling scheme.

6.1.4 Limitations

We have tested the multi-scale hybrid linear model on a wide range of images, with some representative ones shown in Figure 6.15. From our experiments and experience, we observe that the multi-scale hybrid linear model is more suitable than wavelets for representing images with multiple high frequency/entropy regions, such as those with sharp 2-D edges and rich of textures. Wavelets are prone to blur sharp 2-D edges but better at representing low frequency/entropy regions. This probably explains why the hybrid linear model performs slightly worse than wavelets for the Lena and the monarch – the backgrounds of those two images are out of focus so that they do not contain much high frequency/entropy content.

Another limitation of the hybrid-linear model is that it does not perform well on gray-scale images (e.g., the Barbara image, Figure 6.5). For a gray-scale image, the dimension $D$ of a 2 by 2 block is only 4. Such a low dimension is not adequate for any further dimension reduction. If we use a larger block size, say 8 by 8, the block effect will also degrade the performance.

Unlike pre-fixed transformations such as wavelets, our method involves identifying the subspaces and their bases. Computationally, it is more costly. With unoptimized MATLAB codes, the overall model estimation takes 30 seconds to 3 minutes on a Pentium 4 1.8GHz PC depending on the image size and the desired PSNR. The smaller the PSNR, the shorter the running time because the number of blocks needed to be coded in higher levels will be less.
Figure 6.15. A few standard testing images. From the top-left to the bottom-right: monarch (768 × 512), sail (768 × 512), canyon (752 × 512), tiger (480 × 320), street (480 × 320), tree (512 × 768), tissue (microscopic) (1408 × 1664), Lena (512 × 512), earth (satellite) (512 × 512), urban (aerial) (512 × 512), bricks (696 × 648). The multi-scale hybrid linear model out-performs wavelets except for the Lena and the monarch.

6.2 Multi-Scale Hybrid Linear Models in Wavelet Domain

From the discussion in the previous section, we have noticed that wavelets can achieve a better representation for smooth regions and avoid the block artifacts. Therefore, in this section, we will combine the hybrid linear model with the wavelet approach to build multi-scale hybrid linear models in the wavelet domain. For readers who are not familiar with wavelets, we recommend the books of [?].

6.2.1 Imagery Data Vectors in Wavelet Domain

In the wavelet domain, an image is typically transformed into an octave tree of subbands by certain separable wavelet. At each level, the LH, HL, HH subbands contain the information about high frequency edges and the LL subband is further decomposed into subbands at the next level. Figure 6.16 shows the octave tree structure of a level-2 wavelet decomposition. As shown in the Figure 6.17, the vectors \( \{x_i \in \mathbb{R}^D \}_{i=1}^M \) are constructed by stacking the corresponding wavelet
coefficients in the LH, HL, HH subbands. The dimension of the vectors is $D = 3c$ because there are $c$ color channels. One of the reasons for this choice of vectors is because for edges along the same direction, these coefficients are linearly related and reside in a lower dimensional subspace. To see this, let us first assume that the color along an edge is constant. If the edge is along the horizontal, vertical or diagonal direction, there will be an edge in the coefficients in the LH, HL, or HH subband, respectively. The other two subbands will be zero. So the dimension of the imagery data vectors associated with such an edge will be 1. If the edge is not exactly in one of these three directions, there will be an edge in the coefficients of all the three subbands. For example, if the direction of the edge is between the horizontal and diagonal, the amplitude of the coefficients in the LH and HH subbands will be large. The coefficients in the HL subband will be insignificant relative to the coefficients in the other two subbands. So the dimension of the data vectors associated with this edge is approximately 2 (subject to a small error $\epsilon^2$). If the color along an edge is changing, the dimension the subspace will be
higher but generally lower than the ordinal dimension $D = 3c$. Notice that the above scheme is only one of many possible ways in which one may construct the imagery data vector in the wavelet domain. For instance, one may construct the vector using coefficients across different scales. It remains an open question whether such new constructions may lead to even more efficient representations than the one presented here.

### 6.2.2 Estimation of Hybrid Linear Models in Wavelet Domain

In the wavelet domain, there is no need to build a down-sampling pyramid. The multi-level wavelet decomposition already gives a multi-scale structure in the wavelet domain. For example, Figure 6.18 shows the octave three structure of

![Figure 6.18](image_url). The subbands of level-3 bior-4.4 wavelet decomposition of the baboon image.

a level-3 bior-4.4 wavelet transformation of the baboon image. At each level, we may construct the imagery data vectors in the wavelet domain according to the previous section. A hybrid linear model will be identified for the so-obtained vectors at each level. Figure 6.19 shows the segmentation results using the hybrid linear model at three scale levels for the baboon image.

**Vector Energy Constraint at Each Level.** In the nonlinear wavelet approximation, the coefficients which are below an error threshold will be ignored. Similarly in our model, not all the vectors of the imagery data vectors need to be modeled and approximated. We only need to approximate the (coefficient) vectors $\{x_i\}$ that satisfy the following constraint:

$$||x_i||^2 > \epsilon^2.$$  \hspace{1cm} (6.19)

Notice that here we do not need to scale the error tolerance at different levels because the wavelet basis is orthonormal by construction. In practice, the energy of most of the vectors is close to zero. Only a small portion of the vectors at each level need to be modeled (e.g. Figure 6.19).
The overall process of estimating the multi-scale hybrid linear model in the wavelet domain can be summarized as the pseudocode in Algorithm 6.3.

Algorithm 6.3 (Multi-Scale Hybrid Linear Model: Wavelet Domain).

1: function \( \hat{I} = \text{MultiscaleModel}(I, \text{level}, \epsilon^2) \)
2: \( \hat{I} = \text{WaveletTransform}(I, \text{level}); \)
3: for each \( \text{level} \) do
4: \( \hat{I}_{\text{level}} = \text{HybridLinearModel}(\hat{I}_{\text{level}}, \epsilon^2); \)
5: end for
6: \( \hat{I} = \text{InverseWaveletTransform}(\hat{I}, \text{level}); \)
7: return \( \hat{I} \).

6.2.3 Comparison with Other Lossy Representations

In this section, in order to obtain a fair comparison, the experimental setting is the same as that of the spatial domain in the previous section. The experiment is conducted on the same two standard images – the 480 × 320 hill image and the 512 × 512 baboon image shown in Figure 6.10.

The number of levels of the model is also chosen to be 3. In Figure 6.20, the results are compared with several other commonly used image representations including DCT, PCA/KLT, single-scale hybrid linear model and Level-3 biorthogonal 4.4 wavelets (JPEG 2000) as well as the multi-scale hybrid linear model in the spatial domain. The multi-scale hybrid linear model in the wavelet domain achieves better PSNR than that in the spatial domain. Figure 6.21 shows the three recovered images using the same amount of coefficients for wavelets, the hybrid
Figure 6.20. Top: Comparison of several image representations for the hill image. Bottom: Comparison for the baboon image. The multi-scale hybrid linear model in the wavelet domain achieves better PSNR than that in the spatial domain.

linear model in the spatial domain, and that in the wavelet domain, respectively. Figure 6.22 shows the visual comparison with the enlarged bottom-right corners of the images in Figure 6.21.

Notice that in the area around the baboon’s whiskers, the wavelets blur both the whiskers and the subtle details in the background. The multi-scale hybrid linear model (in the spatial domain) preserves the sharp edges around the whiskers but generates slight block artifacts in the relatively smooth background area. The multi-scale hybrid linear model in the wavelet domain successfully eliminates the block artifacts, keeps the sharp edges around the whiskers, and preserves more details than the wavelets in the background. Among the three methods, the multi-scale hybrid linear model in the wavelet domain achieves not only the highest PSNR, but also produces the best visual effect.

As we know from the previous section, the multi-scale hybrid linear model in the spatial domain performs slightly worse than the wavelets for the Lena and monarch images (Figure 6.15). Nevertheless, in the wavelet domain, the multi-scale hybrid linear model can generate very competitive results, as shown in Figure 6.23. The multi-scale hybrid linear model in the wavelet domain achieves better PSNR than the wavelets for the monarch image. For the Lena image, the comparison is mixed and merits further investigation.

6.2.4 Limitations

The above hybrid linear model (in the wavelet domain) does not produce so competitive results for gray-scale images as the dimension of the vector is merely 3 and there is little room for further reduction. For gray-scale images, one may have to choose a slightly larger window in the wavelet domain or to construct the vector using wavelet coefficients across different scales. A thorough investigation of all the possible cases is beyond the scope of this book. The purpose here is to demonstrate (using arguably the simplest cases) the vast potential of a
new spectrum of image representations suggested by combining subspace methods with conventional image representation/approximation schemes. The quest for the more efficient and more compact representations for natural images without doubt will continue as long as the nature of natural images remains a mystery and the mathematical models that we use to represent and approximate images improve.

6.3 Image Segmentation

6.3.1 Hybrid Linear Models for Image Segmentation

Notice that for the purpose of image representation, we normally divide the image $I$ into non-overlapping blocks (see the beginning of Section 6.1.1). The hybrid
linear model fit to the block vectors \( \{ x_i \} \) essentially gives some kind of a segmentation of the image – pixels that belong to blocks in the same subspace are grouped into one segment. However, such a segmentation of the image has some undesirable features. If we choose a very large block size, then there will be severe “block effect” in the resulting segmentation, as all \( b^2 \) pixels of each block are always assigned into the same segment (see Figure 6.5). If we choose a small block size to reduce the block effect, then the block might not contain sufficient neighboring pixels that allow us to reliably extract the local texture.\(^{14}\) Thus, the resulting segmentation will very much be determined primarily by the color of the pixels (in each small block) but not the texture.

One way to resolve the above problems is to choose a block of a reasonable size around each pixel and view the block as a (vector-valued) “label” or “feature”

\(^{14}\)Notice that a smaller block size is ok for compression as long as it can reduce the overhead and subsequently improve the overall compression ratio.
attached to the pixel. In many existing image segmentation methods, the feature (vector) is chosen instead to be the outputs of the block passing through a (pre-fixed) bank of filters (e.g., the Garbor filters). That is, the feature is the block transformed by a set of pre-fixed linear transformations. Subsequently, the image is segmented by grouping pixels that have “similar” features.

From the lessons that we have learned from image representation in the previous section, we observe that the hybrid linear model may be adopted to facilitate this approach of image segmentation in several aspects. Firstly, we can fit directly a hybrid linear model to the un-transformed and un-processed block vectors, without the need of choosing beforehand which filter bank to use. Secondly, the hybrid linear model essentially chooses the linear transformations (or filters) adaptively for different images and different image segments. Thirdly, once the hybrid linear model is identified, there is no further need of introducing a similarity measure for
the features. Feature vectors (and hence pixels) that belong to the same subspace are naturally grouped into the same image segment.

### 6.3.2 Dimension and Size Reduction

Mathematically, identifying such a hybrid linear model for image segmentation is equivalently to that for image representation. However, two things have changed from image representation and make image segmentation a computationally much more challenging problem. First, the number of feature vectors (or blocks) is now always the same as the number of pixels: \( N = WH \), which is larger than that \( (N = WH/b^2) \) in the case of image representation. For a typical \( 512 \times 512 \) image, we have \( N = 31,744 \). Second, the block size \( b \) now can be much larger that in the case of image representation. Thus, the dimension of the block vector \( D = b^2c \) is much higher. For instance, if we choose \( b = 10 \) and \( c = 3 \), then \( D = 300 \). It is impossible to implement the GPCA algorithm on a regular PC for 31,744 vectors in \( \mathbb{R}^{300} \), even if we are looking for up to only four or five subspaces.\(^{15}\)

**Dimension Reduction via Projection.**

To reduce dimension of the data, we rely on the assumption (or belief) that “the feature vectors lie on very low-dimensional subspaces in the high-dimensional ambient space \( \mathbb{R}^D \).” Then based on our discussion in Section 3.2.2, we can project the data into a lower-dimensional space while still being able to preserve the separation of the subspaces. Principal component analysis (PCA) can be recruited for this purpose as the energy of the feature vectors is mostly preserved by their first few principal components. From our experience, in practice it typically suffice to keep up to ten principal components. Symbolically, the process is represented by the following steps:

\[
\{ x_i \} \subset \mathbb{R}^D \xrightarrow{\pi(x_i)} \{ x'_i \} \subset \mathbb{R}^{D'} \xrightarrow{\text{GPCA}} \{ x''_i \} \subset \bigcup_{j=1}^{n} S'_j \subset \mathbb{R}^{D'},
\]

where \( D' \ll D \) and \( i = 1, \ldots, N = WH \).

**Data Size Reduction via Down-Sampling.**

Notice that the number of feature vectors \( N = WH \) might be too large for all the data to be processed together since a regular PC has trouble in performing singular value decomposition (SVD) for tens of thousands of vectors.\(^{16}\) Thus, we have to down sample the data set and identify a hybrid linear model for only a subset of the data. The exact down-sampling scheme can be determined by the user. One can use periodic down-sampling (e.g., every other pixel) or random down-sampling. From our experience, we found periodic down-sampling often

\(^{15}\)The dimension of the Veronese embedding of degree 5 will be in the order of \( 10^{10} \).

\(^{16}\)With the increase of memory and speed of modern computers, we hope this step will soon become unnecessary.
gives visually better segmentation results. The size of the down-sampled subset can be determined by the memory and speed of the computer the user has. Once the hybrid linear model is obtained, we may assign the remaining vectors to their closest subspaces. Of course, in practice, one may run the process on multiple subsets of the data and choose the one which gives the smallest fitting error for all the data. This is very much in the same spirit as the random sample consensus (RANSAC) method. Symbolically, the process is represented by the following steps:

\[
\begin{align*}
\{x_i\} & \xrightarrow{\text{sample}} \{x_i'\} \\
& \xrightarrow{\text{GPCA}} \{x_i'\} \subset \bigcup_{j=1}^n S_j \\
& \xrightarrow{\min d(x_i, S_j)} \{x_i\} \subset \bigcup_{j=1}^n S_j,
\end{align*}
\]

where \(\{x_i'\}\) is a (down-sampled) subset of \(\{x_i\}\).

### 6.3.3 Experiments

Figure 6.24 shows the results of applying the above schemes to the segmentation of some images from the Berkeley image database. A 20 \(\times\) 20 \(\times\) 3 “feature” vector is associated with each pixel that corresponds to the color values in a 20 \(\times\) 20 block. We first apply PCA to project all the feature vectors onto a 6-dimensional subspace. We then apply the GPCA algorithm to further identify subspace-structures of the features in this 6-dimensional space and to segment the pixels to each subspace. The algorithm happens to find three segments for all the images shown below. Different choices in the error tolerance, window size, and color space (HSV or RGB) may affect the segmentation results. Empirically, we find that HSV gives visually better segments for most images.

### 6.4 Image Classification

Notice that images of similar scenes (e.g., landscape, or urban areas, or wild life, etc.) often contain similar textures and characteristics. It is then interesting to see what would be the underlying hybrid linear model for images of each category.

For the example below, we select two groups of eight gray-scale images from the Berkeley segmentation data set [Martin et al., 2001] shown in Figure 6.25. One group contains common natural scenes, and the other contains more structured urban scenes. We randomly sample 100 blocks of 8 by 8 windows from each image,\(^{17}\) stack them into 64-dimensional vectors, and apply our algorithm to obtain a hybrid linear model.

For this example, we preset the desired subspace number as 3, and the algorithm identifies the bases for the subspaces for each group. Figure 6.26 shows all base vectors as 8 by 8 windows. Visually, one can see that these vectors capture the essential difference between natural scenes and urban scenes.

\(^{17}\)The window size and the number of blocks is limited by the computer memory.
Figure 6.24. Image segmentation results obtained from the hybrid linear model. The dimension of the subspace (in homogeneous coordinates) associated with each segment is marked by the number to its right.

Figure 6.25. Top: natural scenes. Down: Urban scenes.

Unlike most of our other examples, such hybrid linear models are “learned” in a supervised fashion as the training images are classified by humans. The so-obtained hybrid linear models can potentially be used to classify new images of natural or urban scenes into one of the two categories.

In principle, we can also treat each image as a single point (in a very high-dimensional space) and fit a hybrid linear model to an ensemble of images. Then different subspaces of the model lead to a segmentation of the images into different categories. Images in the same category form a single linear subspace. We have seen such an example for face images in Chapter 1 (Figure 1.6).
6.5 Bibliographic Notes

*Image Representation and Compression.*

There is a vast amount of literature on finding adaptive bases (or transforms) for signals. Adaptive wavelet transform and adapted wavelet packets have been extensively studied [?, ?, ?]. The idea is to search for an optimal transform from a limited (although large) set of possible transforms. Another approach is to find some universal optimal transform based on the signals [?, ?, ?]. Spatially adapted bases have also been developed such as [?, ?, ?].

The notion of hybrid linear model for image representation is also closely related to the *sparse component analysis*. In [?], the authors have identified a set of non-orthogonal base vectors for natural images such that the representation of the image is sparse (i.e., only a few components are needed to represent each image block). In the work of [?, ?, ?, ?, ?, ?], the main goal is to find a mixture of models such that the signals can be decomposed into multiple models and the overall representation of the signals is sparse.

*Image Segmentation.*

Image segmentation based on local color and texture information extracted from various filter banks has been studied extensively in the computer vision literature (see e.g., [?, ?, ?]). In this chapter, we directly used the unfiltered pixel values of the image. Our segmentation is a byproduct of the global fitting of a hybrid linear model for the entire image. Since the image compression standard JPEG-2000 and the video compression standard MPEG-4 have started to incorporate texture segmentation [?], we expect that the method introduced in this chapter will be useful for developing new image processing techniques that can be beneficial to these new standards.